

AKÇADAĞ MESLEK YÜKSEKOKULU BİLGİSAYAR PROGRAMCILIĞI

2022-2024 ÖNLİSANS BİTİRME PROJESİ

Unity ile tarot uygulaması

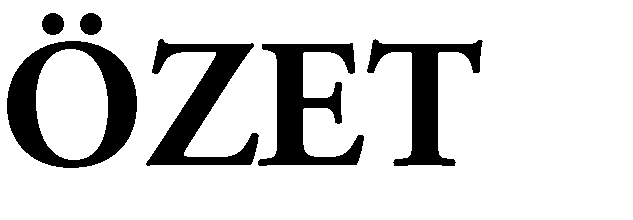
DANIŞMAN

[Öğr. Grv. Cihanşah TULUM](http://akcadag.ozal.edu.tr/cihansah-tulum-2)

Hakkı Tumar

12221215042

2024 MALATYA



Unity uygulaması, tarot kartlarını gösteren basit bir simülasyondur. Kullanıcılar, bir düğmeye tıklayarak kartları karıştırabilir ve rastgele seçilen kartların görüntülerini ve yorumlarını görebilirler. Her bir kart, bir Sprite görüntüsü ve bir yorum metni içerir. Kullanıcılar, arka plan animasyonları veya karmaşık kamera hareketleri olmadan, sadece kartların net bir şekilde görüntülenmesini deneyimlerler. Uygulama, Unity'nin UI ve animasyon yeteneklerini kullanarak geliştirilmiştir ve kullanıcı dostu bir arayüz sunar.

## Projenin Amacı

Bu proje, kullanıcıların tarot kartlarıyla etkileşime geçebileceği ve bu kartların gizemli dünyasını keşfedeceği bir Unity uygulamasıdır. Amacı, kullanıcıların bir tarot destesinden rastgele kartlar seçip bu kartların görsellerini ve içerdikleri sembolizmi incelemelerine olanak tanımaktır. Her kart, kendine özgü bir görsel ve anlam taşır ve kullanıcılar bu kartların sunduğu mesajları keşfederken kendi içsel yolculuklarına çıkabilirler. Uygulama, karmaşık animasyonlardan uzak durarak, odak noktasını kartların sembollerine ve yorumlarına yönlendirir. Bu sayede kullanıcılar, tarot kartlarının derin anlamlarını anlamak ve kendi hayatlarına uygulamak için ideal bir ortam sunar.

**İçindekiler**

[DOĞRULUK BEYANI 4](#_bookmark0)

[TEŞEKKÜR 5](#_bookmark1)

[UNİTY NEDİR 6](#_bookmark2)

[Kullanılan Malzemeler Ve Özellikleri; 7](#_bookmark3)

[KODLAR 9](#_bookmark8)

[KODLARIN AÇIKLAMASI 17](#_bookmark9)

TURGUT ÖZAL ÜNİVERSİTESİ AKÇADAĞ MESLEK YÜKSEKOKULU

# DOĞRULUK BEYANI

Bitirme proje olarak sunduğumuz bu çalışmayı tüm akademik kurallara ve Turgut Özal Üniversitesi Yayın Etiği Komisyonu Yönergesine uygun olarak gerçekleştirdiğimizi ve sunduğumuzu; bu kurallar ve ilkelere aykırı hiç bir yol ve yardıma başvurmaksızın bizzat hazırladığımızı beyan ederiz.

HAKKI TUMAR

TEŞEKKÜR

Bu çalışmanın gerçekleştirilmesinde büyük emeği olan, iki yıl boyunca değerli bilgilerini ve zamanını bizlerle paylaşan, danışman hocam; [Öğr. Grv. Cihanşah TULUM](http://akcadag.ozal.edu.tr/cihansah-tulum-2) ’a teşekkürlerimi sunarım.

HAKKI TUMAR

# **Unity** NEDİR

**Unity Nedir?**

Unity, oyun geliştirme, simülasyonlar ve etkileşimli deneyimler yaratmak için kullanılan güçlü ve esnek bir oyun motorudur. Aşağıda Unity'nin ana özellikleri ve kullanımları özetlenmiştir:

1. **Çok Platform Desteği**: Unity, oyunların ve uygulamaların tek bir kod tabanıyla birçok platformda (PC, konsol, mobil cihazlar, VR/AR cihazları, web vb.) çalışmasını sağlar. Bu, geliştiricilerin oyunlarını geniş bir kitleye ulaştırmasını kolaylaştırır.
2. **Kullanıcı Dostu Arayüz**: Unity, sürükle-bırak özellikleri ve kullanıcı dostu bir arayüz sunar. Bu, hem deneyimli geliştiriciler hem de yeni başlayanlar için kolay kullanım sağlar.
3. **Genişletilebilirlik ve Esneklik**: Unity, C# programlama diliyle özelleştirilebilir ve genişletilebilir. Geliştiriciler, kendi oyun mekaniklerini ve işlevselliklerini kolayca ekleyebilirler.
4. **Güçlü Topluluk ve Kaynaklar**: Unity, geniş bir kullanıcı topluluğuna sahiptir. Eğitim materyalleri, forumlar, dersler ve belgeler gibi birçok kaynak sayesinde geliştiriciler destek alabilir ve öğrenmeye devam edebilir.
5. **Asset Store**: Unity Asset Store, geliştiricilerin projelerinde kullanabilecekleri önceden oluşturulmuş varlıkları (modeller, sesler, araçlar vb.) sunar. Bu, geliştirme sürecini hızlandırır ve kolaylaştırır.
6. **VR ve AR Desteği**: Unity, sanal gerçeklik (VR) ve artırılmış gerçeklik (AR) uygulamaları için güçlü destek sunar. Bu, geliştiricilerin yenilikçi ve etkileşimli deneyimler yaratmasını sağlar.
7. **Gelişmiş Grafik ve Fizik Motoru**: Unity, yüksek kaliteli grafikler ve gerçekçi fizik simülasyonları sunar. Bu, oyunların görsel olarak çekici ve gerçekçi olmasını sağlar.

Unity, oyun geliştiricilerinin ve yaratıcı profesyonellerin projelerini hayata geçirmeleri için kapsamlı ve esnek bir platform sağlar. Hem büyük stüdyolar hem de bağımsız geliştiriciler tarafından yaygın olarak kullanılmaktadır.

# Kullanılan Malzemeler Ve Özellikleri;

1. **Unity Oyun Motoru**
   * **Özellikleri**: Unity, çok platformlu bir oyun motorudur ve hem 2D hem de 3D oyun geliştirme imkanı sağlar. Kullanıcı dostu arayüzü, güçlü grafik ve fizik motorları, genişletilebilirliği ve kapsamlı topluluk desteğiyle öne çıkar.
2. **Tarot Kartı Görselleri (Sprites)**
   * **Özellikleri**: Her biri farklı semboller ve anlamlar taşıyan görsellerden oluşur. Kart görselleri, kullanıcıların tarot kartlarını tanımasını ve yorumlamasını sağlar.
3. **Sprite Renderer Bileşenleri**
   * **Özellikleri**: Kart görsellerini sahnede gösterir. Her kart yuvası için bir sprite renderer bileşeni kullanılır, böylece kartların görselleri ekranda doğru şekilde görüntülenir.
4. **UI Text Bileşeni**
   * **Özellikleri**: Kartların yorumlarını ekranda metin olarak gösterir. Kullanıcıların kartların anlamlarını ve mesajlarını okumasına olanak tanır.
5. **Buton Bileşenleri (Back ve Draw)**
   * **Özellikleri**: Kullanıcı etkileşimini sağlar. "Back" butonu ana sahneye dönmek için kullanılırken, "Draw" butonu kartları karıştırmak ve yeni kartlar çekmek için kullanılır.
6. **C# Programlama Dili**
   * **Özellikleri**: Unity'de oyun mekaniğini ve kullanıcı etkileşimini tanımlamak için kullanılır. Kolay öğrenilebilir ve güçlü bir dil olan C#, projede kartların karıştırılması, görüntülenmesi ve kullanıcı girişlerinin işlenmesi gibi işlevler için kullanılmıştır.

### Ek Malzemeler ve Özellikleri

* **Animasyon Bileşeni** (Projede kullanılmayan)
  + **Özellikleri**: Kart karıştırma ve gösterme işlemleri için animasyonlar eklemek amacıyla kullanılır. Ancak bu projede animasyonlar kullanılmamaktadır.
* **Scene Management Bileşeni**
  + **Özellikleri**: Sahne geçişlerini yönetir. Kullanıcıların ana sahneye geri dönmesini sağlar.

Bu malzemeler ve özellikleri, tarot kartı uygulamanızın işlevselliğini sağlar ve kullanıcıların kartları görüntüleyip yorumlamasına olanak tanır.

### 1. Unity'yi İndirin ve Kurun

Unity Hub'ı indirip kurarak, Unity Editor ve gerekli bileşenleri kurabilirsiniz. Yeni bir proje oluşturmak için Unity Hub'ı kullanabilirsiniz.

### 2. Yeni Bir Proje Oluşturun

Unity Hub'ı açın ve yeni bir 2D veya 3D proje oluşturun. Projeyi oluşturduktan sonra Unity Editor açılacaktır.

### 3. Unity Arayüzünü Anlayın

Unity Editor arayüzü çeşitli panellerden oluşur:

* **Scene**: Oyun sahnenizi tasarladığınız alan.
* **Game**: Oyun oynatılırken görülen ekran.
* **Hierarchy**: Sahnede bulunan tüm oyun nesnelerinin listesi.
* **Inspector**: Seçili oyun nesnesinin bileşenlerini ve özelliklerini düzenleyebileceğiniz alan.
* **Project**: Projenizdeki tüm dosyaları ve varlıkları (assets) içerir.

### 4. Oyun Nesneleri ve Bileşenler

Unity'de her şey bir oyun nesnesidir (GameObject). Bu nesnelere bileşenler ekleyerek işlevsellik kazandırabilirsiniz.

### 5. C# Scriptleri Yazma

Oyun nesnelerine davranış eklemek için C# scriptleri yazılır. Bir script oluşturmak için:

1. **Project** panelinde boş bir alana sağ tıklayın.
2. **Create > C# Script** seçeneğini tıklayın.
3. Scriptinize bir isim verin ve çift tıklayarak Visual Studio veya tercih ettiğiniz kod editöründe açın.

### 6. Temel Bir Script Yazma

İlk scriptinizi yazmak için aşağıdaki adımları izleyin:

#### Örnek: Basit Hareket Scripti

1. **Script Oluşturma**: PlayerMovement adlı bir script oluşturun.
2. **Scripti Açın ve Aşağıdaki Kodu Yazın**:

using UnityEngine;

public class PlayerMovement : MonoBehaviour

{

public float speed = 5f; // Oyuncunun hareket hızı

// Her karede bir kez çağrılan fonksiyon

void Update()

{

// Yatay hareket girişi almak için

float moveHorizontal = Input.GetAxis("Horizontal");

// Dikey hareket girişi almak için

float moveVertical = Input.GetAxis("Vertical");

// Hareket vektörünü oluşturmak için

Vector3 movement = new Vector3(moveHorizontal, 0.0f, moveVertical);

// Hareketi uygulamak için

transform.Translate(movement \* speed \* Time.deltaTime);

}

}

### 7. Scripti Bir Oyun Nesnesine Ekleyin

1. **Hierarchy** panelinde bir oyun nesnesi oluşturun (örneğin, Player).
2. Player nesnesini seçin ve **Inspector** panelinde Add Component butonuna tıklayın.
3. PlayerMovement scriptinizi ekleyin.

### 8. Oyunu Çalıştırın

Unity Editor'de **Play** butonuna basarak oyununuzu çalıştırın. Player nesnesini klavye ok tuşlarıyla hareket ettirebilmelisiniz.

### Özet

* Unity'yi kurun ve yeni bir proje oluşturun.
* Unity arayüzünü ve bileşenleri anlayın.
* C# scriptleri yazmayı öğrenin.
* Scriptleri oyun nesnelerine ekleyin ve test edin.

Unity ile kodlamaya bu temel adımlarla başlayabilirsiniz. Daha karmaşık oyun mekaniği ve etkileşimler için Unity'nin belgelerine ve çevrimiçi derslerine göz atabilirsiniz.

## Adım Adım Tarot Kartı Oyunu Yapımı;

#### 1. Proje Kurulumu

1. **Unity'yi İndirin ve Kurun**: Unity Hub'ı indirip kurun. Unity Hub üzerinden Unity Editor'ü kurun.
2. **Yeni Bir Proje Oluşturun**: Unity Hub'ı açın ve "New" butonuna tıklayarak yeni bir 2D veya 3D proje oluşturun. Projenize uygun bir isim verin ve oluşturun.

#### 2. Temel Oyun Nesneleri ve Bileşenler

1. **Kart Görselleri ve Yorumları İçin Varlıklar (Assets) Ekleyin**:
   * Tarot kartlarının görsellerini (Sprites) ve kartların yorumlarını içeren bir metin dosyasını projenize ekleyin.
   * Bu varlıkları Assets klasörüne ekleyin.
2. **Kart Görsellerini ve Yorumları Tutmak İçin Bir Sınıf Oluşturun**:
   * TarotCard adında bir sınıf oluşturun ve her kartın görselini ve yorumunu tutacak şekilde tasarlayın.
   * TarotCard sınıfı:

csharp

// Tarot kartlarını temsil eden sınıf

[System.Serializable]

public class TarotCard

{ public Sprite image; // Kartın görüntüsü

public string comment; // Kartın yorumu

}

1. **Kartların Gösterileceği Oyun Nesneleri Oluşturun**:
   * Hierarchy panelinde boş bir alana sağ tıklayın ve Create > Empty seçeneğiyle boş bir oyun nesnesi oluşturun. Bu nesneye CardSlots adını verin.
   * CardSlots nesnesi altında her bir kart için bir SpriteRenderer bileşeni ekleyin (örneğin, 3 kart için 3 ayrı SpriteRenderer).
   * Hierarchy panelinde her SpriteRenderer için boş bir oyun nesnesi oluşturun ve bu nesnelere CardSlot1, CardSlot2, CardSlot3 gibi isimler verin. Her birine SpriteRenderer bileşeni ekleyin.
2. **UI Elemanları Ekleyin**:
   * **Button**: Kullanıcının yeni kartlar çekebilmesi için bir buton ekleyin (DrawButton).
   * **Text**: Kartların yorumlarının gösterileceği bir Text bileşeni ekleyin (CardComments).
   * **Button**: Ana sahneye dönmek için bir geri butonu ekleyin (BackButton).

#### 3. Script Yazma

1. **Ana Script Oluşturma**: Scene3 adlı bir C# scripti oluşturun ve aşağıdaki kodu ekleyin:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

using UnityEngine.UI;

// Tarot kartlarını temsil eden sınıf

[System.Serializable]

public class TarotCard

{

public Sprite image; // Kartın görüntüsü

public string comment; // Kartın yorumu

}

// Anahtar script

public class Scene3 : MonoBehaviour

{

// Inspector'dan atanan tarot kartlarının listesi

[SerializeField] List<TarotCard> cards;

// Kartların yerleştirileceği sprite renderer'ların dizisi

[SerializeField] SpriteRenderer[] card\_Slot;

// Kart yorumlarının gösterileceği UI Text bileşeni

[SerializeField] Text card\_Comments;

// Ana sahneye geri dönmek için kullanılan buton

[SerializeField] Button backButton;

// Yeni kart çekmek için kullanılan buton

[SerializeField] Button drawButton;

// Başlangıçta çalışan fonksiyon

void Start()

{

// Ekran yönünü otomatik olarak belirler

Screen.orientation = ScreenOrientation.AutoRotation;

// Buton tıklama dinleyicilerini atar

backButton.onClick.AddListener(() => SceneManager.LoadScene("SampleScene")); // Geri butonuna tıklanınca ana sahneye dön

drawButton.onClick.AddListener(() => Shuffle\_Cards()); // Çekme butonuna tıklanınca kartları karıştır

// İlk karıştırmayı gerçekleştirir

ShuffleCards();

}

// Kartları karıştırma fonksiyonu

public void Shuffle\_Cards()

{

ShuffleCards(); // Kartları karıştırır

}

// Kartları karıştırıp yerleştiren fonksiyon

void ShuffleCards()

{

// Kullanılan indekslerin tutulduğu set

HashSet<int> usedIndices = new HashSet<int>();

string combinedComments = ""; // Bütün yorumları birleştirmek için kullanılan değişken

// Her bir kart yuvası için işlemleri gerçekleştir

for (int i = 0; i < card\_Slot.Length; i++)

{

int rand;

do

{

rand = Random.Range(0, cards.Count); // Rastgele bir indeks seçer

} while (usedIndices.Contains(rand)); // Aynı indeksi tekrar kullanmamak için kontrol eder

usedIndices.Add(rand); // Kullanılan indeksi sete ekler

card\_Slot[i].sprite = cards[rand].image; // Kart yuvasına rastgele seçilen kartın görüntüsünü yerleştirir

combinedComments += cards[rand].comment + "\n"; // Kart yorumunu birleştirir ve yeni satır ekler

}

card\_Comments.text = combinedComments; // Bütün yorumları UI Text bileşenine yerleştirir

}

}

#### 4. Oyun Nesnelerine Script Ekleme

1. Scene3 scriptini CardSlots nesnesine ekleyin.
2. Scene3 scriptinin Inspector panelinde gerekli bileşenleri atayın:
   * Cards: Tarot kartlarını içeren listeyi ayarlayın.
   * Card\_Slot: Kartların gösterileceği SpriteRenderer bileşenlerini atayın.
   * Card\_Comments: Yorumların gösterileceği Text bileşenini atayın.
   * BackButton: Geri butonunu atayın.
   * DrawButton: Kart çekme butonunu atayın.

#### 5. Test Etme ve Son Dokunuşlar

1. **Play** butonuna basarak oyununuzu test edin.
2. Butonlara tıklayarak kartların karıştırıldığını ve yorumların güncellendiğini kontrol edin.

Bu adımlarla tarot kartı uygulamanızı Unity'de başarılı bir şekilde oluşturabilirsiniz. Her adım, temel işlevselliği sağlamak için gereklidir ve proje ihtiyaçlarınıza göre özelleştirilebilir.